Oracle Day 8 – Sub Queries

Note: Please watch my YouTube sessions to better understand the descriptions and queries below

NiC IT Academy YouTube Videos for reference

🔵 Oracle SQL Tutorial - English

https://youtube.com/playlist?list=PLsphD3EpR7F9mmtY2jBt_O8Q9XmvrhQEF

🟡 Oracle SQL - தமிழில்

https://youtube.com/playlist?list=PLsphD3EpR7F-u4Jjp_3fYgLSsKwPPTEH4

📌 Oracle SQL Day wise Video: ENGLISH

Oracle SQL Day 1 – Introduction to Oracle - https://youtu.be/hLnKjYGr730

Oracle SQL Day 2 – SQL Types DDL, DML, DRL, DCL, TCL - https://youtu.be/XpgjXvnfZec

Oracle SQL Day 3 – Constraints in Oracle - https://youtu.be/TmYqeFfHyyc

Oracle SQL Day 4 – SELECT Statements in Oracle - https://youtu.be/tYQfBgUCpoI

Oracle SQL Day 5 – Single Row Functions in Oracle - https://youtu.be/4qJJxQuHLC4

Oracle SQL Day 6 – Joins in Oracle - https://youtu.be/CkaqluC2afE

Oracle SQL Day 7 – Aggregate Functions in Oracle - https://youtu.be/BSiCWzj-py8

Oracle SQL Day 8 – Sub Queries in Oracle - https://youtu.be/KtUCyG2cZe4

Oracle SQL Day 9 – SET Operators in Oracle - https://youtu.be/B0JbGbWsEIA

Oracle SQL Day 10 – Analytical Functions in Oracle - https://youtu.be/gRC3ndWLsoo

Oracle SQL Day 11 - Views in Oracle - https://youtu.be/m8a1UtOmd5k

Oracle SQL Day 12 - Indexes in Oracle - https://youtu.be/reL2O-kvNxc

Oracle SQL Day 13 - Regular Expression - https://youtu.be/k_Eo08vLPhU

Sub Queries in Oracle:

==========

select (select);

2 1

outer Query inner Query

---

select (select);

insert (select);

update (select);

delete (select);

1. Single row sub query

2. Multi row sub Query

3. Multi row multi column subquery

4. Co-related sub query

****

1. Single row sub query:

==========

select (select);

1 row

select max(salary) from employees;

select * from employees where salary =24000;

select * from employees where salary =(select max(salary) from employees);

=

!= or <>

>

>=

<

<=

--select the employees who are getting salary more than the avg salary

select * from employees where salary>(select avg(salary) from employees);

--Some more examples:

==========

--Whose employee job is the same as the job of 'Stephen'?

select * from employees where first_name='Stephen'; -- ST_CLERK

select * from employees where job_id=(

```
select job_id from employees where first_name='Stephen');

select * from employees where first_name='James';

select * from employees where job_id=(
select job_id from employees where first_name='James');
-- ORA-01427: single-row subquery returns more than one row

select * from employees where job_id=(
select distinct job_id from employees where first_name='James');

select * from employees where first_name='Steven';

select * from employees where job_id=(
select distinct job_id from employees where first_name='Steven');
-- ORA-01427: single-row subquery returns more than one row

-- multi row subqueries
select * from employees where job_id IN (
select distinct job_id from employees where first_name='Steven');

-- Whose salary is more than the max salary of the job is “Sales Manager”?


select  * from employees;

select job_id from jobs where job_title='Sales Manager';
```

```
select max(salary) from employees where job_id =(select job_id from jobs where job_title='Sales Manager');

select * from employees where salary >
(select max(salary) from employees where job_id =(select job_id from jobs where job_title='Sales Manager'));
```

-- Whose employee job is same as the job of “Ellen” and who are earning Salary more than “Ellen” salary?

```
SELECT * FROM EMPLOYEES WHERE JOB_ID=
(SELECT job_id FROM EMPLOYEES WHERE first_name='Ellen')
AND SALARY>(SELECT salary FROM EMPLOYEES WHERE first_name='Ellen');
```

-- DISPLAY EMPLOYEES WHO SALARY IS MORE THAN AVERAGE SAL OF DEPARTMENT

?

-- Display senior employee among all the employees?

```
select min(hire_date) from employees;

select * from employees where hire_date = (select min(hire_date) from employees);
```

--Find the second-highest Salary from the Employee table?

```
select max(salary) from employees;
```

-- second highest salary

```
select max(salary) from employees where salary < (select max(salary) from employees);

-- second highest salaried employee

select * from employees where salary =(
select max(salary) from employees where salary < (select max(salary) from employees));

-- Sum of salary of jobs if the sum of salary of jobs are more than the sum of salary of the job is 'CLERK'?

select sum(salary) from employees where job_id like'%CLERK%';

SELECT job_id, SUM(salary)
FROM employees GROUP BY JOB_ID HAVING SUM(SALARY)>
(select sum(salary) from employees where job_id like'%CLERK%');
```


2. Multi row subquery:

```
select * from employees where salary  = (select max(salary) from employees group by department_id);
--ORA-01427: single-row subquery returns more than one row

IN
NOT IN
>ANY
<ANY
>ALL
<ALL
```

--Whose employee job is the same as the job of 'James'?

```
select * from employees where job_id=(
select job_id from employees where first_name='James');
```

--ORA-01427: single-row subquery returns more than one row

```
select * from employees where first_name='James';
```

```
select * from employees where job_id IN (
select job_id from employees where first_name='James');
```

```
select * from employees where salary  IN (select max(salary) from employees group by department_id)
order by department_id;
```

3. Multi column sub query

```
select * from employees where (department_id,salary)  IN (select department_id,max(salary) from
employees group by department_id)
order by department_id;
```

```
select * from employees where department_id > ANY
(select department_id from departments where department_name in('Purchasing','IT','Executive')) order
by department_id;
```

```
select * from employees where department_id > ANY (30,60,90) order by department_id;
```

```
select * from employees where department_id < ANY (30,60,90) order by department_id;

select * from employees where department_id < ALL (30,60,90) order by department_id;

select * from employees where department_id > ALL (30,60,90) order by department_id;
```

-- select the departments where no employees are working

```
select * from departments;

select * from employees;

select * from departments where department_id not in(
select distinct department_id from employees where department_id is not null);
```

---

co-releated sub query

For every one record it execute the inner query

Find employees having atleast one person reporting under him

non co-related sub query:

```
select * from employees where employee_id in (
select manager_id from employees group by manager_id);
```

co-related sub query:

```
select * from employees a where 1 <= (select count(*) from employees b
where b.manager_id=a.employee_id);
```

The Oracle EXISTS operator is a Boolean operator that returns either true or false. The EXISTS operator is often used with a subquery to test for the existence of rows:

An EXISTS condition tests for existence of rows in a subquery. If at least one row returns, it will evaluate as TRUE.

The EXISTS operator returns true if the subquery returns any rows, otherwise, it returns false. In addition, the EXISTS operator terminates the processing of the subquery once the subquery returns the first row.


```
SELECT
  *
FROM
  table_name
  WHERE
    EXISTS(subquery);
```

The EXISTS subquery is used when we want to display all rows where we have a matching column in both tables

The IN clause is faster than EXISTS when the subquery results is very small.

The IN clause can't compare anything with NULL values, but the EXISTS clause can compare everything with NULLs

This returns the employees (in the EMP table) that are managers. It checks for their employee number as a manager (mgr column) and returns them if they are found at least once.

```
select * from employees e1
 where exists ( select null from employees e2
            where e2.manager_id = e1.employee_id);
```